文章编号: 1001－9081(2001)08－0045－02

基于 SNMP 协议的公共机房管理软件设计

段江娇
(厦门大学 计算机系, 福建 厦门 361005)

摘 要: 针对公共机房管理上的要求, 采用软件技术实现了计费管理、权限管理、实时监控等功能。应用软件采用三层结构的形式实现数据库操作, 利用 SNMP 协议作为客户端与数据库访问模块的主要通信协议, 简化了客户端软件设计, 提高了软件的可靠性。

关键词: 公共机房; 管理软件; SNMP 协议
中图分类号: TP311.1 文献标识码: A

1 引言

为了对公共机房上机秩序、访问权限等进行有效的管理及控制, 机房管理人员需要登记上机学生的有关资料、进出机房的时间, 以便进行计费管理, 并且要不定时查看学生的上机情况, 防止使用暴力、黄色软件。管理人员经常来不及处理, 学生意见很大, 机房秩序混乱。为此, 我们提出采用设计专用软件的方法, 辅助机房管理人员进行机房管理, 减轻管理人员的工作。同时, 为逐步实现公共机房的无人管理进行有益的尝试, 为学生提供一个真正自由的上机环境。

2 公共机房管理软件的设计目标

通过对公共机房管理的需求分析, 我们认为公共机房管理软件应具备以下功能:

计费管理 记录学生的上机时间、下机时间, 根据计费方法计算本次上机的费用, 并从学生预交的金额(个人帐户)中扣除。在学生的计算机上实时显示该学生的当前累计上网时长及余额, 当余额不足时, 应提示学生缴费。如果放弃缴费, 则应把计算机锁定, 不允许该帐号在任意计算机上使用。

应该提供完善的费用查询功能, 以便让学生可以核对自己的上机情况。

提供灵活的计费方法, 可对不同类型的学生实行不同的计费方法, 可实现不同时间段的不同计费方法。

权限管理 设定学生使用计算机时所拥有的权限, 例如访问 Internet、运行某个软件等。

实时监控 管理员可以对学生计算机的使用情况进行监视。并可以通过管理台控制学生正在使用的计算机。另一方面, 系统应该记录学生使用计算机的足迹, 如运行什么程序、时间多长等。

客户端软件应具有较强的抗破坏性 客户端软件是安装在学生用的计算机上, 如果该进程被学生强行终止, 将导致无法进行完整的计费管理、权限管理及实时监控等。因此, 客户端软件必须具有较强的抗破坏性, 防止学生采用各种方法对客户端软件进行强行终止及其他破坏。

3 软件设计

3.1 实现方案

3.1.1 软件结构

根据以上分析, 客户端软件必须尽量简单, 而且要可靠运行。为此, 该软件把客户端代理和管理执行模块分开, 以确保客户端代理模块这一关键部分的可靠运行。系统采用三层结构的数据库访问方式, 客户端代理只通过数据库访问模块来实现对数据库的读写操作, 省去了在客户端安装数据库连接程序, 简化数据库访问设置, 如图 1 所示。

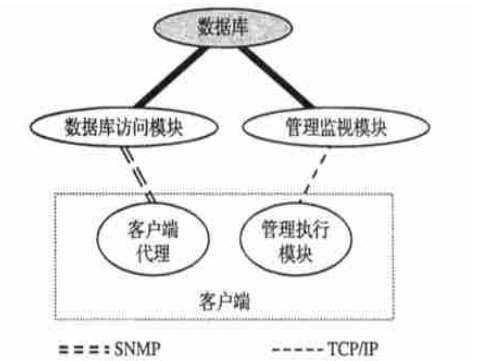

图 1 公共机房管理软件组成

系统采用 MS SQL Server 作为数据库, 集中存储学生基本信息、上机时间、下机时间、本次上机费用、学生交款记录、当前余额、学生上网足迹等信息。

在学生用的计算机上运行客户端代理及管理执行模块。其功能是提供登录窗口, 锁定/放开计算机, 显示最新余额; 接收管理监视模块发送来的命令, 如终止某进程、调整音量等控制命令, 并进行相应的处理, 向管理监视模块返回处理结果; 向数据库访问模块发送命令, 如请求查询上机记录等, 并等待返回处理结果等。

在管理员计算机上运行管理监视模块。其职能是对学生的基本信息、上机费用进行增删改等管理操作; 向各代理发出命令, 如查看学生用的计算机上当前的活动进程, 并接收代理模块返回的信息。

数据库访问模块接收代理发出的命令, 进行分析处理, 如执行数据库查询、更新, 并将结果送给代理。

3.1.2 通信协议的选择

客户端代理收集计算机运行信息, 如是否处于登录状态等, 执行相对重要的任务。因此, 客户端代理必须实现隐身进

作者简介: 段江娇, 女, 山西临汾人, 讲师, 主要研究方向: 计算机网络技术及应用、计算机软件技术及应用.

?1994-2014 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

程,以防被破坏。因此,我们采用 SNMP 协议作为客户端代理与数据库访问模块之间的通信协议,并把客户端代理按照标准的 SNMP 代理结构编写成 DLL 形式,该 DLL 由客户端计算机上 SNMP 服务调用。

管理监视模块与管理执行模块之间采用 TCP/IP 协议的 Socket 直接通信。

3.2 管理监视/执行模块

管理执行模块接收管理监视模块的命令,其结构如图 2 所示。

管理监视模块实现学生计算机的屏幕远程抓取及控制。另外一个主要功能是完成软件系统的日常维护,如编辑上机学生资料、增加上机款额、计费方法设置、各种统计报表等。

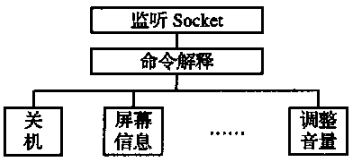

图 2 管理执行模块

3.3 基于 SNMP 协议的客户端代理

客户端代理模块包括用户信息窗口及消息处理模块,如图 3。

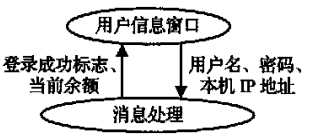

图 3 客户端代理模块组成

客户端代理模块采用 SNMP 应用程序的扩展代理方式。扩展代理实际上是一个 DLL 文件,一般安装在 Windows98 的%SystemRoot% \SYSTEM 目录下,通过读取注册键,SNMP 服务在启动后可以找到扩展代理的安装位置,并启动之。

客户端代理通过 SNMP 的基本操作 GetResponse、GetRequest、SetRequest、Trap 等进行数据交换。设计流程描述如下:

用户输入用户名、密码后,监视线程唤醒扩展代理,形成一个 SNMP Trap 报文,并向 SNMP 服务模块发送该 Trap。SNMP 服务模块收到 Trap 报文,分解出数据元素,并进行相应的数据库操作,然后通过 SNMP GetResponse 报文把处理结果(登录成功与否,出错原因、用户当前余额)送给代理。登录成功后,监视线程每隔一定时间发出一个 Trap 事件,请求用户当前的最新余额信息。

3.4 基于 SNMP 协议的数据库访问

如图 4,SNMP 服务模块接收各个代理发送来的数据请求。同时,也可以使用 GetRequest 命令向各个代理请求数据。

数据库访问模块统一完成数据库操作,如密码验证、更新余额信息、查询余额信息。

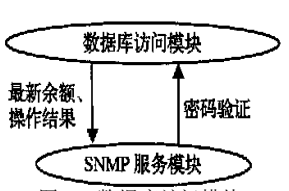

图 4 数据库访问模块

4 关键代码实现

客户端代理主要的程序代码如下:

```
//与 Windows 的 SNMP 服务接口的标准函数
HANDLE ghTrapEvent;                                    //陷入事件句柄
HANDLE ghPollThread;                                   //监视线程句柄
BOOL WINAPI SnmpExtensionInit (DWORD
dwTimeZeroReference, HANDLE *phPollForTrapEvent,
AsnObjectIdentifier *pSupportedView)
{
  BOOL fResult = TRUE;
  *pSupportedView = NULL;
  *phPollForTrapEvent =
    CreateEvent(NULL, FALSE, FALSE, NULL);
  if( *phPollForTrapEvent != NULL){
    ghTrapEvent = *phPollForTrapEvent;
    ghPollThread = CreateThread ( NULL, 0,
      (LPTHREAD_START_ROUTINE)MonitorThread,
      0, 0, &ghMonitorThreadId);
    if(ghPollThread == NULL)
      fResult = FAISE;
  }
  else
    fResult = FAISE;
  return (fResult);
}

//形成陷入报文
BOOL WINAPI SnmpExtensionTrap
  (AsnObjectIdentifier *pEnterprise, AsnInteger *pdwGenericTrap,
  AsnInteger *pdwSpecificTrap, AsnTimeTicks *pdwTimeStamp,
  RFC11578VarBindList *pVariableBindings)
{
  static BOOL fCleanUp = FAISE;
  if(fCleanUp == FALSE){
    *pEnterprise = NULL;
    *pdwGenericTrap = 'SNMP-GENERICTRAP-ENTERSPECIFIC'
    *pdwSpecificTrap = gdwSpecificTrap;
    *pVariableBindings -> list = VarBindList.List;
    *pVariableBindings -> len = VarBindList.len;
    fCleanUp = TRUE;
  }
  else{
    if( VarBindList.List != (RFC1157VarBind *) NULL){
      VarBindList.len = 0;
      VarBindList.List = (RFC1157VarBind *) NULL;
    }
    fCleanUp = FALSE;
  }
  return (fCleanUp);
}
```

SNMP 服务模块的主要代码是陷入的接收、处理,如下:

```
#include< mgmtapi.h>
void main (void){
HANDLE hTrapEvent;
AsnObjectIdentifier TrapEnterprise;
AsnIPAddress TrapIpAddress;
RFC1157VarBindList TrapVarBinds;
```

(下转第 49 页)

?1994-2014 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

特别是前面提到的回调函数的注册,因此基本不需添加更多的代码。上层调用回调函数时是直接的,只要按照 DDK 中的说明让每个回调函数完成它应该完成的任务就行了。而物理中断的处理则与实际的 ADSL 板卡有关。下面说明使用的中断处理策略。

板卡的逻辑控制由 FPGA 完成,FPGA 只在上一次 DMA 结束后才产生下一次中断。我们正是利用这一点来保证只在上一次 DMA 结束后才进行下一次 DMA 和/或将上一次收 DMA 得到的数据放入内部循环缓冲。

1) 一旦发生中断,ADSLControl::OnHardwareInt()被调用,通过读 FPGA 端口寄存器确定中断类型。如果是收发中断,收中断优先于发中断处理。然后调用 ADSLControl::IntHandler()。

2) ADSLControl::IntHandler()根据中断类型标志调用相应的 Comx 的中断处理函数。

3) 如果是"物理信号丢失中断",Comx::SigLostIntHanlder()被调用。Comx::SigLostIntHandler()向上发出事件通知,告诉上层"modem 状态寄存器中的 ACE-RLSD 位已被清除",然后关中断并关闭物理线路。

4) 如果是"收中断",Comx::RecvIntHandler()被调用。Comx::RecvIntHandler()先把上一次收 DMA 得到的数据从物理连续缓冲送入收循环缓冲,然后启动下一次收 DMA。

5) 如果是"发中断",Comx::SendIntHandler()被调用。Comx::SendIntHandler()先把可能存在的上一次收 DMA 得到的数据从物理连续缓冲送入收循环缓冲中,然后在必要时启动发 DMA。

最后,为安装此驱动,模仿 serial.inf 写了 adslport.inf 文件。在安装时使用"控制面板"中的"添加新硬件",不让 Windows 选择新硬件,而是在列表中选择"端口(COM & LPT)"。之后,点击"从磁盘安装",并指定 Adslport.inf 文件。然后,在安装了"标准 33600bps 调制解调器"(指定使用的端口为前面刚安装的端口)后,就能通过拨号网络来上网了。

5　总结

为了在 Windows 95/98 下使用我们自己设计的 ADSL 卡上网,采用虚拟一个串口以及与该串口相连的标准话带 modem 的方法,利用 VToolsD 驱动程序开发包编写了一个"伪串口"port driver。

最终的实验结果证明了上述方法的有效性。在 290MHz 的 CPU 加 64MB 内存的机器上,低层(包括所有的协议头)数据传输速率为 635KB/s 左右,而去掉所有的协议头后的数据传输速率是 565KB/s 左右。这些数据是通过 FTP 一个连续的大文件时测得的。

由于呈现在用户面前的是一个标准话带 modem,所以其使用方法简洁明了,且具有相当的通用性。话带 modem 能胜任的场合,此系统亦能胜任,且具有大得多的数据吞吐量。

从 vxd 的编写角度来说,这种"伪串口"实现方式充分利用了 Win95 既有的串口通信结构具有的方便性,极大地减少了所需编制的代码量,而效果是很好的。

参考文献

[1] Walter Oney. System Programming for Windows95[M]. Microsoft Press, 1996.

[2] Charles A. Mirho, Andre Terrisse. Windows95 通信编程[M]. 贺军,高胜友,贺民,等译. 北京:清华大学出版社,1996.

(上接第 46 页)

```
If (SnmpMgrTrapListen( &hTrapEvent) == FALSE)
狖
  DWORD dwError = GetLastError();
  If (dwError != SNMP_MGMTAPI_TRAP_DUPINIT)
  狖
    //错误
  狅
狅
else 狖                                    //检测到陷入事件
  DWORD dwResult;
  TrapVarBinds.list = (RFC1157VarBind *)
    SnmpUtilMemAlloc(sizeof(RFC1157VarBind));

  dwResult = WaitForSingleObject(hTrapEvent, INFINITE);
  If (dwResult != WAIT_FAILED) 狖
    If(dwResult != WAIT_TIMEOUT)狖
      AsnInteger TrapGeneric;
      AsnInteger TrapSpecific;
      AsnTimeticks TrapTimeStamp;
      While (SnmpMgrGetTrap (&TrapEnterprise, &TrapIpAddress,
        &TrapGeneric, &TrapSpecific, &TrapTimeStamp,
        &TrapVarBinds) == TRUE )
      狖
        //对接收到的陷入数据(TrapVarBinds)进行处理
      狅
    狅
    if(GetLastError() != SNMP_MGMTAPI_NOTRAPS)
      //Error Handling
    狅
  狅
  ResetEvent(hTrapEvent);
  SnmpUtilOidFree( &TrapEnterprise);
  SnmpUtilVarBindListFree( &TrapVarBinds);
狅
```

5　结论

考虑到公共机房管理上的特殊需求,本文采用三层数据库访问模式的软件结构,并把客户端中的关键处理部分设计成 SNMP 代理的形式,确保该进程的安全可靠运行,不被强行终止。系统通过统一模块访问数据库,而客户端采用标准的 SNMP 协议、TCP/IP 协议通信,这样即使客户端的操作系统不同时,只要能支持 SNMP 协议均可连接到该系统中。因而系统具有良好的灵活性及可扩展性。

参考文献

[1] 周明天,汪文勇. TCP/IP 网络原理与技术[M]. 北京:清华大学出版社,1997.

[2] 岑贤道,安常青. 网络管理协议及应用开发[M]. 北京:清华大学出版社,1998.

?1994-2014 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net